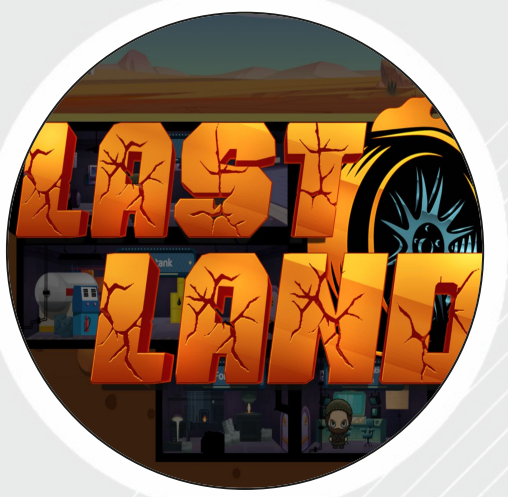
SPYWOLF

Security Audit Report

(TESTNET)



Completed on

October 11, 2023

@SPYWOLFNETWORK
@SPYWOLFNETWORK
SPYWOLF.CO

OVERVIEW

This audit has been prepared for **Last Land** to review the main aspects of the project to help investors make make an informative decision during their research process.

You will find a a summarized review of the following key points:

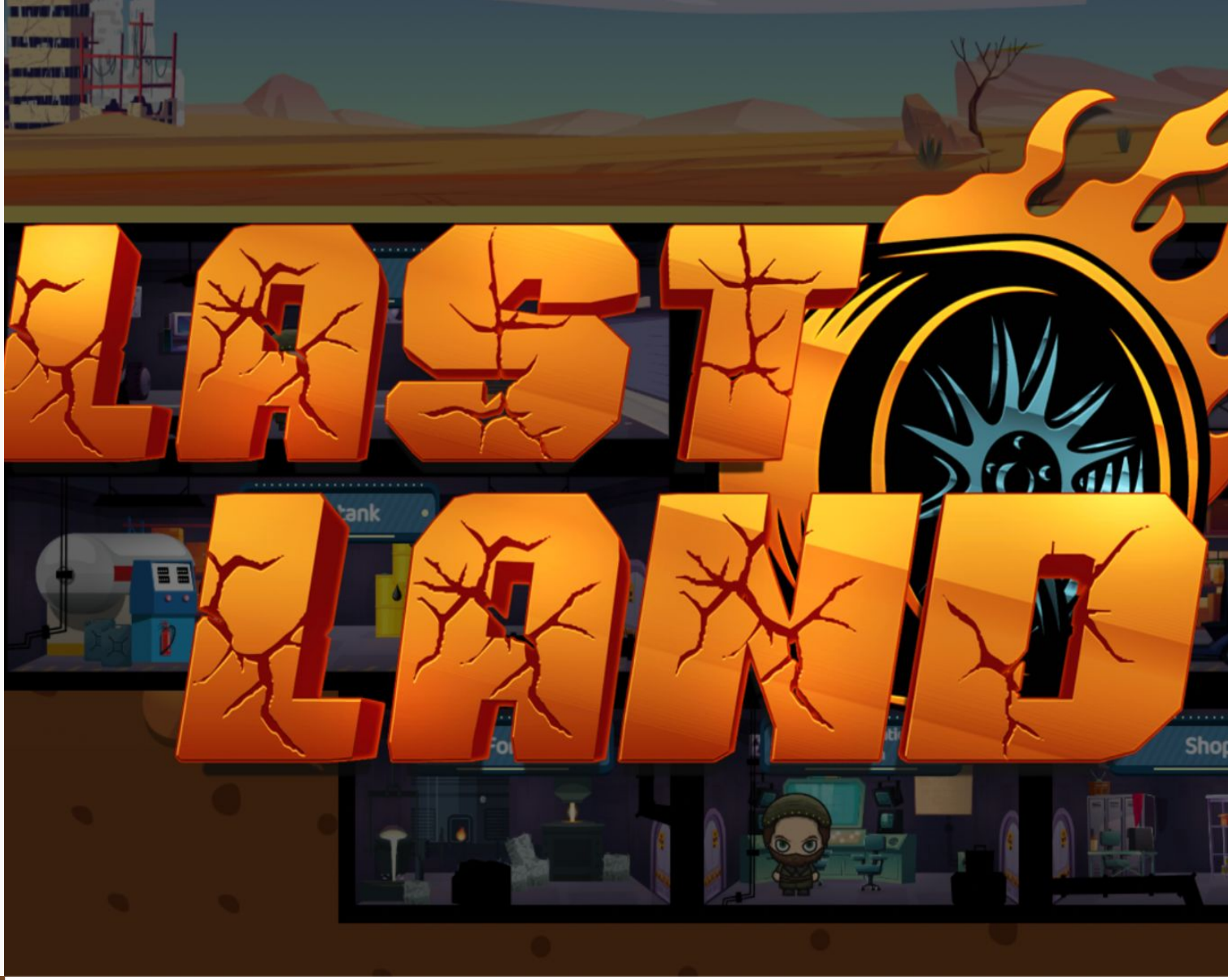
- ✔ Contract's source code
- ✔ Owners' wallets
- ✔ Tokenomics
- ✔ Team transparency and goals
- ✔ Website's age, code, security and UX
- ✔ Whitepaper and roadmap
- ✔ Social media & online presence

The results of this audit are purely based on the team's evaluation and does not guarantee nor reflect the projects outcome and goal

- SPYWOLF Team -

TABLE OF CONTENTS

Last Land



PROJECT DESCRIPTION

According to their website:

"In the aftermath of the apocalypse, LastLand is your final sanctuary. As a survivor, your mission is clear: build and manage your shelter, explore the wastelands, and gather the vital metal resources needed for survival.

Upgrade your cars and gas tanks to extend their wasteland expeditions, and even refine metal to ensure a steady supply. The highlight? Earn BNB with an astounding APY of up to 1100%."

For more info: https://docs.lastland.io

Release Date: TBD

Category: Staking

CONTRACT INFO

Token Name
N/A

Symbol
N/A

Contract Address
N/A

Network
N/A

Language
Solidity

Deployment Date
N/A

Contract Type
Staking

Total Supply
N/A

Status
Not aunched

TAXES

Buy Tax
10%

Sell Tax
none

*Taxes cannot be changed

Our Contract Review Process

The contract review process pays special attention to the following:

- Testing the smart contracts against both common and uncommon vulnerabilities
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line-by-line manual review of the entire codebase by industry experts.

Blockchain security tools used:

- OpenZeppelin
- Mythril
- Solidity Compiler
- Hardhat

TOKEN TRANSFERS STATS

Transfer Count	N/A
Uniq Senders	N/A
Uniq Receivers	N/A
Total Amount	N/A
Median Transfer Amount	N/A
Average Transfer Amount	N/A
First transfer date	N/A
Last transfer date	N/A
Days token transferred	N/A

SMART CONTRACT STATS

Calls Count	N/A
External calls	N/A
Internal calls	N/A
Transactions count	N/A
Uniq Callers	N/A
Days contract called	N/A
Last transaction time	N/A
Created	N/A
Create TX	N/A
Creator	N/A

VULNERABILITY CHECK

Design Logic	Passed
Compiler warnings.	Passed
Private user data leaks	Passed
Timestamp dependence	Passed
Integer overflow and underflow	Passed
Race conditions and reentrancy. Cross-function race conditions	Passed
Possible delays in data delivery	Passed
Oracle calls	Passed
Front running	Passed
DoS with Revert	Passed
DoS with block gas limit	Passed
Methods execution permissions	Passed
Economy model	Passed
Impact of the exchange rate on the logic	Passed
Malicious Event log	Passed
Scoping and declarations	Passed
Uninitialized storage pointers	Passed
Arithmetic accuracy	Passed
Cross-function race conditions	Passed
Safe Zeppelin module	Passed
Fallback function security	Passed

THREAT LEVELS

When performing smart contract audits, our specialists look for known vulnerabilities as well as logical and access control issues within the code. The exploitation of these issues by malicious actors may cause serious financial damage to projects that failed to get an audit in time. We categorize these vulnerabilities by the following levels:

High Risk

Issues on this level are critical to the smart contract's performance/functionality and should be fixed before moving to a live environment.

Medium Risk

Issues on this level are critical to the smart contract's performance/functionality and should be fixed before moving to a live environment.

Low Risk

Issues on this level are minor details and warning that can remain unfixed.

Informational

Information level is to offer suggestions for improvement of efficacy or security for features with a risk free factor.

FOUND THREATS

⚠ High Risk

No high risk-level threats found in this contract.

⚠ Medium Risk

No medium risk-level threats found in this contract.

⚠ Low Risk

No low risk-level threats found in this contract.

Informational

Users can buy coins with bnb.
10% from the buy value will withdrawn from the contract to the project owners (fees).

```
function buyCoins(address referrer) public payable {
    require(isLaunched(), "contract hasn`t launched yet");
    require(msg.sender == tx.origin, "not allowed");
    require(msg.value >= MIN_PURCHASE,'insufficient amount');
    require(referrer != msg.sender, "wrong referrer");

    User storage user = users[msg.sender];

    payFee(msg.value);
    recordReferral(referrer,msg.sender);
    payRefFee(msg.sender,msg.value);

    if(user.stats.totalDeposited == 0) {
        totalUsers = totalUsers.add(1);
    }

    uint256 coinsAmount = fromBnbToCoins(msg.value);

    user.balance.coins = user.balance.coins.add(coinsAmount);
    user.stats.totalDeposited = user.stats.totalDeposited.add(msg.value);

    totalStaked = totalStaked.add(msg.value);

    emit coinsBought(coinsAmount, msg.sender, block.timestamp);

}

function payFee(uint256 amount) internal {

    uint256 marketingFee = amount.mul(MARKETING_FEE).div(PERCENTS_DIVIDER);
    uint256 devFee = amount.mul(DEV_FEE).div(PERCENTS_DIVIDER);

    payable(marketingFund).transfer(marketingFee);
    payable(dev).transfer(devFee);

}
```

Informational

Players can reinvest their harvested metal and convert it into coins.
When users do this from the buyCoinsForMetal function, players will receive additional 5% coins bonus.
On each reinvestment event 10% fee of reinvested amount is paid from the contract's current balances to project owners.

```
uint256 constant public PERCENTS_DIVIDER = 10_000;
uint256 constant public REINVESTMENT_BONUS = 500; //5%
function buyCoinsForMetal(uint256 amount) public {
    require(users[msg.sender].balance.metal >= amount, "amount exceeds balance");

    uint256 coins = fromMetalToCoins(amount);
    uint256 bonusCoins = coins.mul(REINVESTMENT_BONUS).div(PERCENTS_DIVIDER);

    users[msg.sender].balance.metal = users[msg.sender].balance.metal.sub(amount);
    users[msg.sender].balance.coins = users[msg.sender].balance.coins.add(coins).add(bonusCoins);

    uint256 bnbAmount = fromCoinsToBnb(coins);

    payFee(bnbAmount);

    emit coinsBought(coins.add(bonusCoins), msg.sender, block.timestamp);
}
```

Informational

Users can sell their harvested metal for bnb in ratio 1:100000.
100000 Metal = 1 BNB

```
uint256 constant public METAL_PRICE = 0.00001 ether; // 1 BNB = 100000 METAL

function fromMetalToBnb(uint256 amount) public pure returns(uint256) {
    return amount.mul(METAL_PRICE);
}

function sellMetal(uint256 amount) public {
    require(users[msg.sender].balance.metal >= amount, "amount exceeds balance");

    uint256 payout = fromMetalToBnb(amount);

    users[msg.sender].balance.metal = users[msg.sender].balance.metal.sub(amount);
    users[msg.sender].stats.totalWithdrawn = users[msg.sender].stats.totalWithdrawn.add(payout);


    payable(msg.sender).transfer(payout);

    emit metalSold(msg.sender,amount,block.timestamp);

}
```

Informational

Users can use the forgeMetal function to stake their harvested metal and get daily metal ROI of 0.8%.

```
function forgeMetal(uint256 amount) public {
    require(amount > 0, "wrong amount");
    require(users[msg.sender].balance.metal >= amount, "insufficient amount");

    users[msg.sender].balance.metal = users[msg.sender].balance.metal.sub(amount);

    deposit(amount,msg.sender);
}

function deposit(uint256 amount, address userAddr) internal {
    syncForge(userAddr);

    forges[userAddr].depositedAmount = forges[userAddr].depositedAmount.add(amount);
    forges[userAddr].checkpoint = block.timestamp;

    emit onDeposit(userAddr,amount,block.timestamp);
}

function syncForge(address userAddr) internal {
    uint256 dividends = getForgeDividends(userAddr);

    if(dividends > 0) {
        forges[userAddr].harvestableAmount = forges[userAddr].harvestableAmount.add(dividends);
    }
}

function getForgeDividends(address userAddr) public view returns(uint256 totalAmount){
    Forge memory forge = forges[userAddr];
    uint256 share = forge.depositedAmount.mul(DAILY_ROI).div(PERCENT_DIVIDER);

    uint256 from = forge.checkpoint;
    uint256 to = block.timestamp;

    totalAmount = share.mul(to.sub(from)).div(TIME_STEP);

}
```

Informational

Users can harvest their forged metal via the harvestForgedMetal function. Harvested metal will be added to their metal balances.

```
function harvestForgedMetal() public {
    uint256 harvestedAmount = harvest(msg.sender);

    users[msg.sender].balance.metal = users[msg.sender].balance.metal.add(harvestedAmount);
}

function harvest(address userAddr) internal returns(uint256){
    uint256 dividends = getForgeDividends(userAddr);
    uint256 claimableAmount = getClaimableAmount(userAddr);
    uint256 harvestableAmount = forges[userAddr].harvestableAmount;

    require(dividends.add(claimableAmount).add(harvestableAmount) > 0, "nothing to harvest");

    forges[userAddr].checkpoint = block.timestamp;

    if(harvestableAmount > 0) {
        dividends = dividends.add(harvestableAmount);
        forges[userAddr].harvestableAmount = 0;
    }

    if(claimableAmount > 0) {
        uint256 unstakedAmount = unstake(userAddr);
        dividends = dividends.add(unstakedAmount);
    }


    emit onHarvest(userAddr, dividends, block.timestamp);

    return dividends;
}
```

Informational

Users can use coins to buy new cars and upgrade their fuel tanks to endure more time in the wastelands.

```
function buyCar(uint8 _type) public {
    require(_type > 0 && _type < 8, "wrong car type");

    Car memory car = getCarByType(_type);
    User storage user = users[msg.sender];

    require(user.balance.coins >= car.cost, "insufficient coins amount");

    user.balance.coins = user.balance.coins.sub(car.cost);
    user.cars.push(car);

    totalCars = totalCars.add(1);

    emit carBought(_type, msg.sender, block.timestamp);
}
```

```
function upgradeFuelTank() public {

    User storage user = users[msg.sender];

    require(user.stats.totalDeposited != 0, "user not found");
    require(user.fuelTankLevel < 2, "fuel tank has max lvl");

    FuelTank memory fuelTank = getFuelTankByLvl(user.fuelTankLevel.add(1));

    require(user.balance.coins >= fuelTank.cost, "insufficient coins balance");

    user.balance.coins = user.balance.coins.sub(fuelTank.cost);
    user.fuelTankLevel = user.fuelTankLevel.add(1);

    emit fuelTankUpgraded(msg.sender,user.fuelTankLevel,block.timestamp);
}

function getFuelTankByLvl(uint256 lvl) public pure returns(FuelTank memory) {
    if(lvl == 0) {
        return FuelTank(0, 4 hours);
    } else if(lvl == 1) {
        return FuelTank(150, 6 hours);
    } else {
        return FuelTank(300, 8 hours);
    }
}
```

RECOMMENDATIONS FOR

GOOD PRACTICES

1. Consider fundamental tradeoffs
2. Be attentive to blockchain properties
3. Ensure careful rollouts
4. Keep contracts simple
5. Stay up to date and track development

LastLand

GOOD PRACTICES FOUND

- ✔ The owner cannot stop or pause the contract
- ✔ The smart contract utilizes "SafeMath" to prevent overflows

This is **R**eturn **O**f **I**nvestment (**ROI**) dapp.
Player's ROI depends on how frequently they put their cars into 'wastelands' mode to collect metal.

There are 3 type of assets in the contract's logic:
BNB, Coins and Metal

Users can purchase coins with BNB.
With Coins users can purchase cars, equipment, metal and upgrade their fuel tank.
With Metal users can choose to either convert and withdraw to BNB or purchase Coins to buy new cars and equipment.

Users can choose between 7 different cars.
Each car have different ROI per hour starting from 1.92% for cheapest car up to 2.5% for the most expensive car (without added equipment).

Users can choose between 13 different levels of equipment which will increase their cars ROI.

If all 12 levels of equipment is purchased, ROI of each car increases up to +20% from the current ROI so it becomes 2.3% ROI for the cheapest car and 3% ROI for the most expensive car with equipment added.

Users can choose to claim their metal and sell it for BNB or forge it for additional 0.8% ROI in Metal.

There is referral system with up to 11% bonus for each newly invited user with upline of 3 referrals. In order to get the bonus, the one who invites new users must be already invested in the contract, otherwise he/she won't receive the bonus. Referral rewards are distributed as 50% coins and 50% metal.

1st ref in the upline gets 7% bonus
2nd ref in the upline gets 3% bonus
3rd ref in the upline gets 1% bonus

Example:
User who invites new investor will receive 7% of invested value if the new investor points it as his referral. User's referral will get 3% and the User's referral's referral will get 1%.

ROI dapps can be volatile and risky.

TOKENOMICS

SPYWOLF

CRYPTO SECURITY

Audits | KYCs | dApps
Contract Development

ABOUT US

We are a growing crypto security agency offering audits, KYCs and consulting services for some of the top names in the crypto industry.

- **OVER 150 SUCCESSFUL CLIENTS**
- **MORE THAN 500 SCAMS EXPOSED**
- **MILLIONS SAVED IN POTENTIAL FRAUD**
- **PARTNERSHIPS WITH TOP LAUNCHPADS, INFLUENCERS AND CRYPTO PROJECTS**
- **CONSTANTLY BUILDING TOOLS TO HELP INVESTORS DO BETTER RESEARCH**

To hire us, reach out to
contact@spywolf.co or
t.me/joe_SpyWolf

FIND US ONLINE

SPYWOLF.CO

SPYWOLF.NETWORK

@SPYWOLFNETWORK

@SPYWOLFOFFICIAL

@SPYWOLFNETWORK

@SPYWOLFNETWORK

Disclaimer

This report shows findings based on our limited project analysis, following good industry practice from the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, overall social media and website presence and team transparency details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report.
While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the disclaimer below – please make sure to read it in full.

DISCLAIMER:
By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any and all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis, and does not constitute investment advice.
No one shall have any right to rely on the report or its contents, and SpyWolf and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (SpyWolf) owe no duty of care towards you or any other person, nor does SpyWolf make any warranty or representation to any person on the accuracy or completeness of the report.
The report is provided "as is", without any conditions, warranties or other terms of any kind except as set out in this disclaimer, and SpyWolf hereby excludes all representations, warranties, conditions and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report. Except and only to the extent that it is prohibited by law, SpyWolf hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against SpyWolf, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of use of this report, and any reliance on this report. The analysis of the security is purely based on the smart contracts, website, social media and team.
No applications were reviewed for security. No product code has been reviewed.